2019

A Deep Learning Approach to Diagnosing Schizophrenia

Justin Barry
University of Central Florida

A DEEP LEARNING APPROACH TO DIAGNOSING SCHIZOPRHENIA

by

JUSTIN BARRY
B.S. Christopher Newport University, 2011

A thesis submitted in partial fulfilment of the requirements
for the degree of Master of Science
in the Department of Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Spring Term
2019

ABSTRACT

In this article, the investigators present a new method using a deep learning approach to diagnose schizophrenia. In the experiment presented, the investigators have used a secondary dataset provided by National Institutes of Health. The aforementioned experimentation involves analyzing this dataset for existence of schizophrenia using traditional machine learning approaches such as logistic regression, support vector machine, and random forest. This is followed by application of deep learning techniques using three hidden layers in the model. The results obtained indicate that deep learning provides state-of-the-art accuracy in diagnosing schizophrenia. Based on these observations, there is a possibility that deep learning may provide a paradigm shift in diagnosing schizophrenia.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

The contents of this entire thesis were previously published here [30].

Machine learning has been applied to a variety of applications, this includes computer vision applications [3] and medical diagnostics [20, 23]. Typically, all machine learning algorithms need to be provided with significant features from data to learn the patterns and perform classification [25, 13]. Deep learning is a subfield of machine learning which can extract features and perform classification on its own [36, 2]. Recently deep learning has gained a lot of interest in the diagnosis of Schizophrenia (SCZ).

Schizophrenia is a mental disorder primarily affecting people between the ages of 16 and 30. Its positive features include hallucinations, delusions, psychosis; its negative features include impaired motivation and social withdrawal [26]. SCZ usually manifests in adolescence or early twenties and continues to worsen into adulthood. An at-risk phase, called the prodromal phase, often precedes the full-blown disorder, though there have been cases of sudden onset in previously healthy individuals. SCZ's effects are severe. Gone untreated, SCZ places the burden of care on the individuals family. Thus, there is a social incentive to efficiently and accurately diagnose SCZ. Additionally, research has shown that early diagnosis of the disease can reduce treatment costs [24], thus increasing the need for a reliable diagnosing mechanism. Currently, diagnosing SCZ involves subjective analysis of a patients test results and mental history, though symptom overlap with other mental disorders [26] can occur, increasing the risk of misdiagnosis. An automated and efficient, physiology-based diagnosis of SCZ would be beneficial. Currently, there are no well-established biomarkers for identifying SCZ, though studies have shown [18, 16] that effective biomarkers for SCZ exist. This paper shows how multimodal MRI data can be used to classify SCZ.

Magnetic Resonance Imaging (MRI) is a non-invasive imaging modality that returns valuable in-

formation about the physiology of the human brain, including size, shape, and tissue structure [4]. MRI captures either structural or functional information. Functional MRI (fMRI) utilizes Blood-oxygen-level-dependent (BOLD) signals to capture an approximate measurement of activity between remote regions in the brain [11]. Structural MRI (sMRI) provides information on varying characteristics of brain tissue such as gray matter, white matter, and cerebrospinal fluid [34]. The challenge with using sMRI data to diagnose based on structural changes brought on by SCZ is the overlap in structural change brought on by factors closely linked with SCZ such as alcoholism and anti-psychosis medication [4]. Previous studies have shown that the combination of fMRI and sMRI data can be used in conjunction with a deep learning autoencoder to classify mental disorders including SCZ [27, 29, 37]. In one such study [27], researchers used an autoencoder, 4-layers deep in encoding and decoding, to learn the features of the input data, then used SVM to classify the data with 92% accuracy.

Problem Statement

From the multimodal features derived from the brain magnetic resonance imaging (MRI) scans, we aim to automatically diagnose subjects with Schizophrenia. Two modalities of MRI scans are used to obtain these features: functional and structural MRI. Given a set of training data with these features and corresponding labels (either Schizophrenic or not), the goal is to build a classifier that is specific enough to accurately diagnose with as little false-positives and false-negatives[1] as possible and generic enough such that it is robust to any small perturbations in the future (test) data.

[1]Here, in a probabilistic classifier, a false-positive is seen as the classifier generating a 'high' value (say > 0.7) for a subject, whereas, in reality, the subject doesn't have SCZ. The threshold can typically be determined by empirical analysis. We chose a probabilistic output instead of a clear binary output to enable to end-user/radiologist to make an informed and collective decision, rather than have the algorithm make a firm decision.

Dataset

Data on 144 test subjects came from a 3T Siemens Trio MRI scanner (12-channel head coil). There were 75 controls and 69 patients with SCZ [29]. All data was collected at the Mind Research Network[2]. Data was preprocessed to distill independent components using the independent component analysis results from a study found in [1, 28], and leveraging spatiotemporal regression to mitigate the potential of low bias or high variance [12].

The dataset contains the following:

- **Training FNC:** FNC features for the training set. These are correlation values. They describe the connection level between pairs of brain maps over time.
- **Training SBM:** SBM features for the training set. These are standardized weights. They describe the expression level of ICA brain maps derived from gray-matter concentration.
- **Training Labels:** Labels for the training set. The labels are indicated in the "Class" column. 0 = 'Healthy Control', 1 = 'Schizophrenic Patient'.
- **Testing FNC:** FNC features for the test set. Test subject labels have been removed. Your task is to predict these unknown labels from the provided features.
- **Testing SBM:** SBM features for the test set. Test subject labels have been removed.

This is a classic (probabilistic) binary classification problem. Here, we want to find a prediction model $\hat{f} : \mathcal{X} \rightarrow \hat{y}$, where $\hat{y} = \hat{f}(x)$ is a class prediction for any given observation x such that $\hat{y} \in [0.0, 1.0]$. These observations are d-dimensional vectors where each dimension d represents a

[2]funded by a Center of Biomedical Research Excellence (COBRE) grant 5P20RR021938/P20GM103472 from the NIH to Dr. Vince Calhoun - [7]

particular feature of the MRI dataset. The resulting prediction probability for a given observation gives the severity of SCZ a subject has. The greater the value of $\hat{y}$, the more likely the subject has SCZ.

Contribution

The contribution of our work is three-fold. First, we apply a series of traditional (and popular) machine learning algorithms to our problem that perform not only well, but also provide insight into the classification mechanisms. This lays the groundwork for future experiments and presents a good yardstick to measure the efficacy of any new algorithms working with this dataset. Second, the features selected were significant in deciding whether a given subject was Schizophrenic or not. To validate our proposed method and our deep network's internal architecture, we ran multiple, automated trials of our model using different sets of hyperparameters. Third, the proposed architecture handily outperforms the traditional methods accuracies while maintaining an acceptable level of generality. The related results further confirm the effectiveness of the proposed model for Schizophrenic subject classification.

CHAPTER 2: TRADITIONAL MACHINE LEARNING APPROACHE

We explore a few well-known machine learning models to predict if a test subject has schizophrenia or not. We restrict ourselves to three basic algorithms, namely, logistic regression, support vector machine (SVM) and random forest (an ensemble technique). These (and their variants) are some of the most commonly used algorithms both in the industry and academia. A brief overview of these algorithms are given below and detailed parameters for the experimental setup are also given.

Logistic Regression

Logistic regression is a variation of linear regression where the output (dependent) variable is binary (categorical variable with two values such as 'yes' or 'no') rather than a continuous variable [17].

Multivariable problems are frequently encountered in medical research. Researchers are typically faced with questions such as "What is the relationship of one or more exposure variables (x's) to a disease or illness outcome (y). If we consider a dichotomous disease outcome with 0 representing *not diseased* and 1 representing *diseased*, and if the illness/disease is coronary heart disease (CHD) status, then the subjects would be classified as either 0 ("without CHD") or 1 ("with CHD"). Suppose, that the researcher is interested in a single dichotomous exposure variable, for instance, obesity status, classified as "yes" or "no" (or it could be a continuous variable as 'obese value (BMI)' and could take on some predefined bucketed values such as '0-5', '5-10' or '25-35' etc). In such situations, the research question translates to finding the extent to which obesity is associated with CHD status.

The factors that could potentially contribute to an illness/disease represents a collection called

independent variables and the variable that needs to be predicted (the outcome) is called *dependent* variable. More generally, the independent variables are denoted as $\mathbf{X} = \{x_1, x_2, \ldots, x_n\}$ where n is the number of variables/factors being considered and x is a When y, the outcome, is dependent on (or related to) a number of x's, then it is a *multivariable* problem.

Logistic regression is a mathematical modeling approach that can be used to describe the relationship of several x's to a *dichotomous* dependent outcome. Like linear regression, it assumes a linear relationship between the predictor and output variables. It is used when one wants to study whether some event occurred or not such as this loan will be paid back, or it won't; customer booked a deal or not, soccer-team A will win the game etc. The input variables may not be continuous.

The *logistic function*, which describes the mathematical form on which the logistic model is based, is given as:

$$f(z) = \frac{1}{1 + e^{-z}} \tag{2.1}$$

As the value of z varies from $-\infty$ to $+\infty$, the value of $f(z)$ takes on values from 0 to 1. The fact that the logistic function $f(z)$ ranges between 0 and 1 is the primary reason the logistic model is so popular. This model is designed to describe a probability, which is always some number between 0 and 1. In medical terms, such a probability gives the *risk* of an individual getting a disease.

More formally, the output of logistic regression is a predictor variable that gives the probability of an event happening. The model is given by:

$$\log\left(\frac{p(x)}{1 - p(x)}\right) = \beta_0 + x\beta_1 \tag{2.2}$$

Solving for p, this gives

$$p(x) = \frac{e^{\beta_0 + x\beta_1}}{1 + e^{\beta_0 + x\beta_1}} = \frac{1}{1 + e^{-(\beta_0 + x\beta_1)}} \tag{2.3}$$

where β_0 is the bias or intercept term and β_1 is the coefficient for the single input value x.

The ability to model the odds has made the logistic regression model a popular method of statistical analysis. The logistic regression model can be used for prospective, retrospective, or cross-sectional data. More details on its origins, applications and advantages can be seen in [10].

Some key parameters used to run logistic regression are given below:

1. C = 25.0. This can be tweaked for optimal regularization. The larger values gives more freedom to the model. Here, in order to contain any large increases or decreases in the coefficient values due to small perturbations in the data, the inverse of the regularization parameter is used.
2. For multinomial loss, the 'sag'[1] solver is used.
3. L2 regularization is used to improve generalization performance, i.e., the performance on new, unseen data. L2 was chosen (as against L1 or others) is because the dataset is not sparse and is better at generalizing the model complexity.

Support Vector Machine

Support Vector Machines (SVM) are a class of statistical models first developed in the mid-1960s by Vladimir Vapnik and became popular in 1992 (then introduced again by Boser, Guyon and Vapnik in [5]). Over the years, it has evolved considerably into one of the most flexible and effective machine learning tools available [9, 33] and [32] provides a comprehensive treatment of SVMs.

[1]Stochastic Average Gradient Descent: It allows to train a model much faster that other solvers when the data is very large. To use it effectively, the features must be scaled.

SVMs are used for both classification and regression. It operates by maximizing the margin (hyperplane separating the datasets) from either sides of the dataset. This boils down to a quadratic optimization problem where one minimizes the following equation

$$\Phi(w) = 1/2||W||^2 + C\sum_{i=1}^{n}\xi_i \tag{2.4}$$

subject to $y_i(w^T x_i + b) \geq 1 - \xi_i$. C is the tradeoff parameter between error and margin and $\xi_i \geq 0$.

The general idea is that the original feature space can always be mapped to some higher-dimensional feature space where the training data set is separable by a hyperplane. $C > 0$ is the penalty parameter of the error term. There are different kernels that can be applied here. Some of the basic kernels are the following:

- linear
- polynomial
- radial basis function (RBF)
- sigmoid

In general, the RBF kernel is a reasonable first choice. It nonlinearly maps samples into a higher dimensional space and so, it can handle cases where the relation between the predictor variables and the class labels is nonlinear.

Random Forest

Random Forests (RF) are a popular ensemble techique used to build predictive models for both classification and regression problems. It is among the most successful classifiers not only because of their accuracy, but also, due to their robustiness and ability to work with skewed datasets across a variety of domains. A study by M. Fernandez-Felgado [14], evaluated 179 classifiers from 17 different families and concluded that random forests were the best performing classifier among these families. Furthermore, their performance was significantly better than that of others.

Ensemble methods use multiple 'base' learning models to make the final prediction. In the case of random forest, it generates many unpruned decision trees[2], forming a forest of random, uncorrelated decision trees to arrive at the best possible answer. RF uses bagging to generate bootstrap samples of the training dataset and uses CART method to build trees. Each tree casts a vote for the classification of a new sample and the proportion of the votes in each class across the set of trees will be decided as the predicted probability vector.

Assuming that the training set is

$$D = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n) \tag{2.5}$$

drawn randomly from a (possibly unknown) probability distribution $(\mathbf{x}_i, y_i) \sim (X, Y)$. The goal is to build a classifier which predicts y from $\mathbf{x}$ based on the data set of examples D.

Given an ensemble of (possibly weak) classifiers $h = h_1(\mathbf{x}), \ldots, h_k(\mathbf{x})$, if each $h_k(\mathbf{x})$ is a decision tree, then the ensemble is a random forest. We define the parameters of the decision tree for classifier $h_k(\mathbf{x})$ to be $\Phi_k = (\phi_{k1}, \phi_{k2}, \ldots, \phi_{k,p})$. These parameters include the structure of the tree,

[2]Also called *weak learners* that has a minimum accuracy of > 0.5.

which variables are split in which node, splitting crietria etc. Typically, this is written as

$$h_k(\mathbf{x}) = h(\mathbf{x}|\Phi_k) \tag{2.6}$$

which implies that the decision tree k leads to a classifier $h_k(\mathbf{x}) = h(\mathbf{x}|\Phi_k)$. Here, it is important to note that a feature f_i appears in node k of the j^{th} tree is randomly chosen from a model variable Φ. For the final classification $f(\mathbf{x})$ (which combines the classifiers $h_k(\mathbf{x})$), each tree casts a vote for the most popular class at input $\mathbf{x}$, and the class with the most votes wins. More specifically, given dataset $D = (\mathbf{x}_i, y_i)_{i=1}^n$, we train a family of classifiers. Each classifier $h_k(\mathbf{x}) \equiv h(\mathbf{x}|\Phi_k)$ is, in our case, a predictor. The outcome is $y = \{0, 1\}$, representing a pair of unique decisions (in our case, non-Schizophrenic and Schizophrenic patient). For generalizations and more detailed treatment of RFs, refer to [6].

Feature Selection

An important step in machine learning is feature selection. For any type of prediction or classification model, only the relevant and most useful features must be used. It helps in creating a model with better accuracy while requiring less data (by removing irrelevant and redundant features). In fact, if feature selection isn't done, then the resulting model will likely become more complex and less accurate. We used several techniques to determine feature importance and eventually, the mean of all the importance ranks. The techniques used were *Randomized Lasso, Recursive Feature Elimination* and *Random Forest*.

Figure 2.1 show the rankings of all the 411 features. A threshold was manually decided by looking at this figure and deciding a cut-off rank (0.35). This enabled us to choose all features with importance greater than 0.35 and thus enable the classifier to not only provide better accuracy, but

also, run faster with less data. The first ten features sorted according to their importance is given in Table 2.1. The selected features (with rank-threshold = 0.35) are:

['FNC295', 'FNC244', 'SBM_map67', 'FNC302', 'SBM_map61', 'FNC226', 'FNC289', 'FNC220', 'FNC33', 'FNC243', 'SBM_map36', 'FNC183', 'FNC37', 'FNC48', 'FNC78', 'FNC61', 'FNC297', 'FNC333', 'SBM_map7', 'FNC290', 'SBM_map75', 'FNC208', 'FNC292', 'FNC40', 'SBM_map17', 'FNC265', 'FNC171', 'FNC189', 'FNC353', 'FNC62', 'FNC185', 'FNC13', 'FNC337', 'FNC5', 'FNC30', 'FNC68', 'FNC150', 'FNC211', 'FNC293', 'FNC328', 'FNC89', 'FNC106', 'FNC165', 'FNC221', 'SBM_map64', 'FNC75', 'FNC83', 'FNC29', 'FNC102', 'FNC142', 'FNC194', 'FNC200', 'FNC210', 'FNC219', 'FNC256', 'FNC279', 'FNC304', 'SBM_map72']

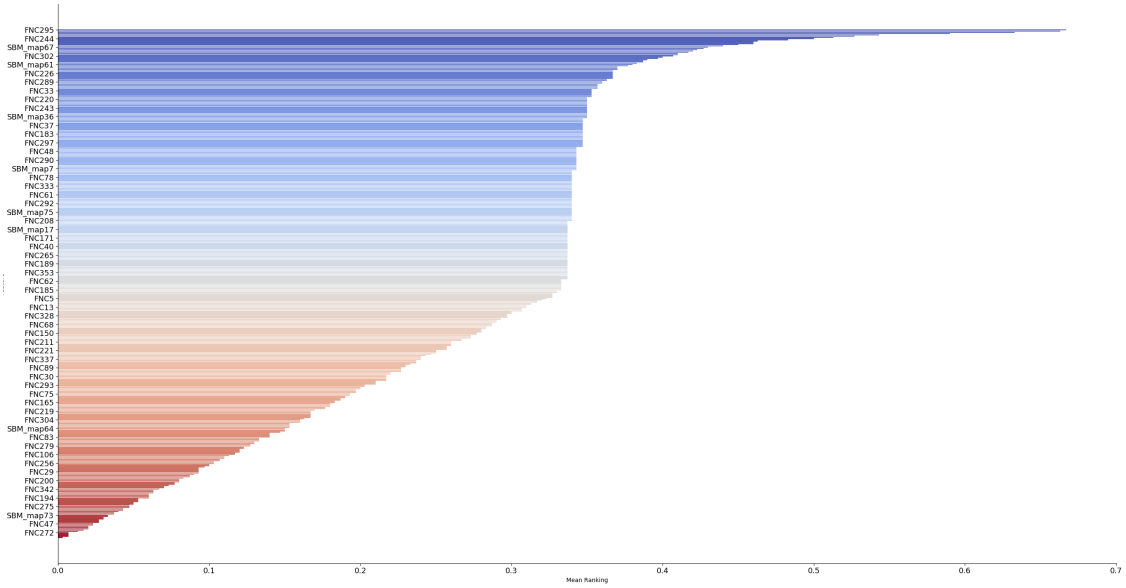

Figure 2.1: Feature rankings.*

* Only 80 features are shown on the x axis for readability purposes.

Table 2.1: The first ten feature rankings from the feature selection step.

Feature	RFE	RForest	Mean
FNC244	1	1	0.667
FNC295	1	0.95	0.65
SBM_map67	1	0.83	0.61
FNC302	1	0.75	0.583
SBM_map61	1	0.65	0.55
FNC226	1	0.57	0.523
FNC33	1	0.52	0.507
FNC289	1	0.51	0.503
FNC183	1	0.41	0.47

CHAPTER 3: APPLICATION OF DEEP LEARNING BINARY CLASSIFIER

Deep learning [22] is a set of machine learning algorithms that model high-level abstractions in data using architectures consisting of multiple nonlinear transformations. Based on artificial neural networks, deep neural networks aim to learn a given domain as a nested hierarchy of concepts, where each concept is defined based on simpler concepts. In a deep neural network, each neuron in each layer represents one aspect of the domain it tries to learn, and in totality, it aims to learn a full representation of the input. Each node also has a weight that represents the strength of its relationship with the output. As the model is fed more and more data, the strengths of these relationships either reinforces or declines and this adjustment continues until training stops.

A major advantage of deep learning is that one neither needs to identify the significant features nor have to do feature engineering. The architecture learns the features incrementally on its own. This eliminates the need for domain expertise and explicit feature-extraction and feature-engineering[1]. Another advantage is that it tends to solve a problem end-to-end whereas, in traditional machine learning approach, the problem has to be broken down into smaller, more-manageable pieces and solved with a variety of statistical and probabilistic approaches.

A clear disadvantage of deep learning (as of this writing) is the inability of the architecture to clearly explain itself. It provides very little justification as to the importances of features and what type of feature-engineering happened behind the scenes. This lack of interpretability is a big reason why many sectors in business have not yet adopted deep learning. In contrast, traditional machine learning approaches like linear regression, logistic regression, SVMs, and random forests all pro-

[1]We say this with caution. Researchers still do not have a clear idea how it learns features, what (good and bad things) it learns, how fast it learns and if it forgets anything.

vide a clear explanation of why and how it made a given decision. In addition to this advantage, deep learning requires a very large amount of data to train and computationally very intensive. Hence, the need for expensive GPUs and large infrastructure to support them. Furthermore, it has yet to establish a robust, theoretical foundation, which leads to its next disadvantage, the difficulty of determining the topology, training mechanism, and hyperparameters of a model. The lack of theory and guidance makes building a good deep learning model less of an engineering feat and more of an artistic endeavor. As there is little theory for guidance (as of this date), building a good deep learning architecture is an art, possibly resulting in a uninterpretable model.

Keras[8] is a high-level neural-networks API, written in python and capable of running on top of Tensorflow, CNTK or Theano. It was developed with a focus on enabling fast experimentation. Keras was used here to diagnose if a patient was schizophrenic or not.

Architecture

The architecture used was rather a simple one. The input layer has 411 neurons, denoted as *dimensions*, which are the 411 features of the data set. Note that we do not used the curated 55 features obtained through feature selection. Instead, we pass the full dimensionality of the data set to the model and allow it to decide which features are the most relevant. The first hidden layer has 512 neurons, each with a *Rectified Linear Unit*[2] activation function. The output layer (the second layer) consisted of just one neuron with a *sigmoid* activation function. Sigmoid is used to ensure that the resulting prediction probabilities fall between 0.0 and 1.0. The loss function used was binary-crossentropy and the optimizer used was 'adam' (Adaptive Moment Estimator) - a stochastic, first-order gradient-based optimzer that can be used instead of the classical stochastic

[2]ReLU is a non-linear function which is essentially a half-wave rectifier $f(z) = max(z, 0)$. ReLU typically learns faster than $tanh(z)$ or $1/1 + e^{-z}$ when there are many layers present in the network.

grdient descent procedure to update network weights. Finally, the model was compiled using the 'accuracy' metric. This function is used to judge the performance of the model.

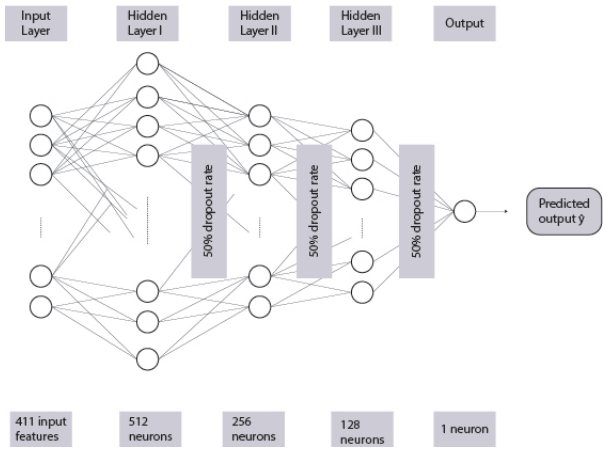

Figure 3.1: Neural network architecture diagram.

Before the training data was fed to the deep neural network model, it was scaled appropriately using a standard scaler. This standardizes the feature values by removing the mean and scaling to unit variance. We chose a set of initial values for the weights to help the model achieve better and faster convergence. This was achieved by the Xavier-Glorot method [15]. This helps in mitigating the well-known problems in back-propagation algorithms, that of, vanishing gradients and exploding gradients (though known to occur very rarely).

Deep neural networks usually contain millions of parameters and hence, it is imperative to attempt to reduce the parameter space as much as possible. In this context, we make such attempts by regularizing the weights. It is well-known that the quality of the regularization method significantly affects both the discriminative power and generalization ability of the trained models. To achieve an acceptable level of generalization, we apply a dropout technique that randomly sets certain neurons' responses to zero with a given probability. In our case, we provide a 50% dropout rate., which means that half of the neurons in any given layer, would output a zero. This technque, though

simple in theory and implementation, has far-reaching benefits not only in helping to generalize the mode, but also, in vastly improving the speed of learning, particularly for very deep networks. This is due to the fact that training batches updates only a subset of all the neuron at a time and conveniently avoid co-adaptation of the learned feature representations [19].

We augment the deep architecture, at every layer, with an application of an aggressive dropout regularization. This boosts the capability fo the network to learn in a generalized way and avoids over-fitting [31].

CHAPTER 4: RESULTS AND DISCUSSION

The dataset used to train the model was obtained from combining the correlation values between the pairs of brain maps (the FNC dataset) and the standardized weights that describe the ICA brain maps (SBM dataset). This combined feature vector was concatenated with their corresponding training labels to create a full set of feature vectors. This was used to train various ML and deep learning models.

We first discuss the results obtained from running traditional machine learning approaches. Logistic regression, when run with 5000 iterations, $L2$-penalization and 'sag' solver, produced an accuracy score 0.8277. A more detailed parameter listing is given below whose cross-validation score, when run with the above parameters is 0.8277.

Table 4.1: Logistic Regression Parameters

Parameter	Value
C	25.0
class_weight	None
dual	False
fit_intercept	True
intercept_scaling	1
max_iter	5000
multi_class	ovr
n_jobs	1
penalty	12
random_state	None
solver	sag
tol	0.0001
verbose	0
warm_start	False

SVM, when run with un-optimized parameters, achieved a lesser accuracy of 0.7988. We did not perform hyper-parameter optimization here. It wouldn't surprise us, if SVM was run with proper hyper-parameter settings, gives better results. SVM was used in our context to do binary classification with an *rbf* kernel and it produced a good model with a cross-validation score of 0.8268. We chose RBF kernel because the number of hyperparameters is much less than that of polynomial kernel. A detailed list of parameters are in Table 4.2:

Table 4.2: SVM Parameters

Parameter	Value
cache_size	200
class_weight	None
coef0	0.0
decision_function_shape	ovr
degree	3
gamma	0.1
kernel	rbf
max_iter	-1
probability	False
random_state	None
shrinking	True
tol	0.001
verbose	False

A few aspects we did not experiment with are the value of C and γ. Ideally, a good (C, γ) has to be identified to produce a good classifier. C (the soft-margin cost function) helps in controlling the influence of each individual support vector and γ controls the variance. A large γ may lead to high bias and low variance models and vice-versa.

Experiments with Random Forest with 5000 trees yielded a very good model with a cross-validation score of 0.8333 and its respective parameter list is given in Table 4.3:

Table 4.3: Random Forest Parameters

Parameter	Value
bootstrap	True
class_weight	None
criterion	gini
max_depth	None
max_features	auto
max_leaf_nodes	None
min_impurity_decrease	0.0
min_impurity_split	None
min_samples_leaf	1
min_samples_split	2
min_weight_fraction_leaf	0.0
n_estimators	5000
n_jobs	-1
oob_score	False
random_state	42
warm_start	False

Table 4.4: Cross-Validation accuracy scores from various models

ML Algorithm	Accuracy
Logistic Regression	0.8277
SVM	0.8268
Random Forest	0.8333
Deep Learning	0.9444

In our deep learning experiments, the total trainable parameters, weights and biases combined, were 375,297 as given in Table 4.5. This number is equal to the summation of the number of trainable parameters in all layers. Let $n^{(L)}$ be the number of neurons in layer L of the network, where $n^{(0)}$ is the number of input features. We can calculate the number of trainable parameters in layer L as $n^{(L)} \times (n^{(L-1)} + 1)$, where the additional 1 is for the bias unit in layer $L - 1$. Then the number of trainable parameters for layer 1 is $n^{(1)} \times (n^{(0)} + 1) = 512 \times (411 + 1) = 210,944$, as shown in Table 4.5. The total number of trainable parameters for the network is the summation of

the number of trainable parameters per layer and is given as $210,944 + 131,328 + 32,896 + 129 = 375,297$.

Table 4.5: Dimensions at each layer of deep learning architecture

Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 512)	210944
dropout_1 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 256)	131328
dropout_1 (Dropout)	(None, 256)	0
dense_3 (Dense)	(None, 128)	32896
dense_4 (Dense)	(None, 1)	129

Total params: 375,297.0
Trainable params: 375,297.0
Non-trainable params: 0.0

The dropout mechanisms applied in each layer significantly reduced the computation time and improved generalization. We trained several deep learning models with different number of hidden layers, dropout, activation functions with the training data. After considerable number of experiments, we zoned on a depth of three hidden layers, number of nodes in each hidden layer (512, 256 and 128) and dropout rate (of 50%). We then ran this general framework on the training dataset with different batch sizes and epochs. Each run on the training dataset was also evaluated by a validation set (of 20%) to validate the effectiveness of the classifier.

The final model's results are visualized in Fig 4.1. The accuracy for the training set stays more or less at 100% while that of the testing set is close, reaching a steady state at 94.444%. And, this result was achieved at the 19^{th} epoch and 45^{th} batch. The best loss (computed by binary-crossentropy) for the training dataset was $4.5488e-06$, while that of the validation set was less than 0.28. During the other stages of training, the accuracy and loss, though was promising, did not reach to this level.

During experiments with various epochs and batch-sizes, though the accuracy tends to be close to 1.0, the validation loss tends to vary and rather higher than 0.25. It is to be noted that there is a fine balance between number of epochs and batch-sizes. Typically, a larger batch size is preferred for good normalization. The greater the mini-batch size, the lesser will be the variance between each mini-batch. Smaller batch sizes typically results in finer gradient descent steps, leading to higher latency in convergence. Larger batch sizes will lead to executing less gradient descent steps and hence the need to train on more epochs. In most cases though, the accuracy will not differ drastically.

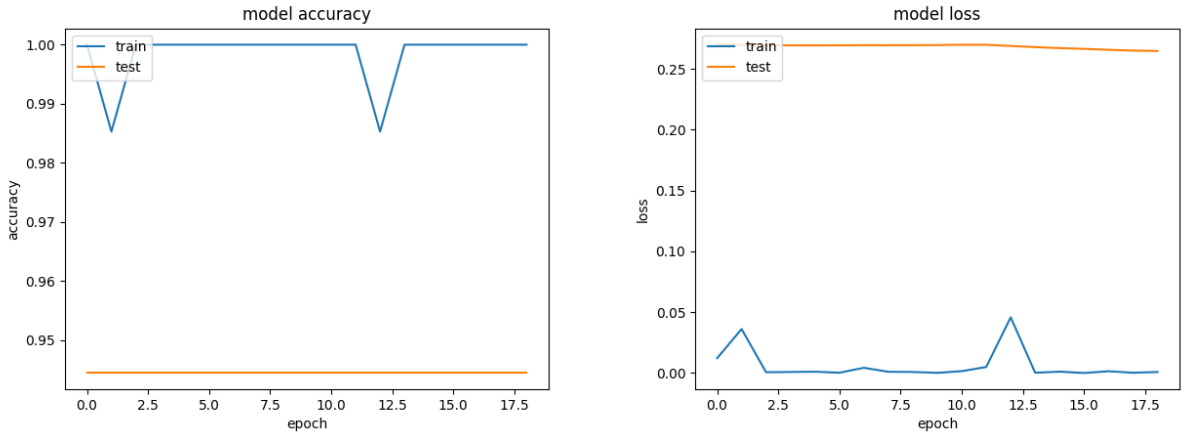

Figure 4.1: Accuracy and loss plots for batch-size = 45 and number-of-epochs = 19.

Limitations

This work has limitations from several fronts. An immediately noticeable one is the size of the training dataset, consisting *only* 86 observations as against to the test dataset that has 119,748 observations. This deficiency in training data will directly impact any classifier's performance. However, we did not find class-imbalance[1]. Out of the given training dataset, we aren't sure about

[1] 40 observations out of 86 had a class of '1', which is a good 46.51% of the training set.

the distribution of the kinds of observations (i.e., the distribution of the relationships among the features). It would have been nice to have orthogonal feature values among the observations and the types of relationships., which would result in a rich set of data, despite having a low caridinality of the training set. In addition to these data issues, the lack of labels for the test dataset prevented us from performing several more statistical analysis of the model's performance (like ROC curve, confusion matrix, F-1 score etc).

On the modeling front, particularly when building deep learning architecture, again, the small size of the training dataset prevented us from exploring many options in hyper-parameter tuning. For experiments with deep neural networks, one typically needs a very large set of training data. This not only allows the investigator to search the parameter space in a fairly comprehensive way[2], but also enable the investigator to generalize the model well (else, there is a risk of overfitting). We had to settle with rather uncomfortable numbers for epochs and batch-size, to name a few. Recent research in training deep neural networks with very little training set (called 'One-Shot Learning') looks promising [21, 35].

[2]Randomized search as against to Grid Search and exhaustive search.

CHAPTER 5: CONCLUSIONS AND FUTURE WORK

In this experimental paper, we attempt to use deep learning techniques to perform classification of patients as Schizophrenic or not based on fMRI data. We have used a deep learning architecture with three hidden layers and achieve reasonably good classification accuracy as observed in the validation-accuracy and loss results. We observed, in a consistent way, that the features selected (in other words, the ranking of features) were indeed significant in deciding whether a given subject is Schizophrenic or not. Our experiments with completely different sets of features (those below the threshold of 0.35) did not achieve a level of accuracy even close to what we attained by performing rigorous feature selection. In some cases, we observed up to a 28% increase in validation accuracy when we explicitly chose those features that were above the 0.35 threshold.

Our next research objective is to discover new features and use more powerful techniques of deep-learning to gain higher accuracy. Some areas we intend to cover are unsupervised feature learning for feature-engineering and pretrain initial weights, while acknowledging that this exercise would be complex and not easily verifiable. We shall also explore converting MRI data into 2D images and perform feature learning before applying convolutional methods on it. We plan to use autoencoders on top of CNNs to better learn features.

In the context of the deep-learning architecture itself, we plan to treat certain neurons in a prefential way. Currently, in our experiments, we used standard dropout, rather than assigning dropout rates for specific sets of neurons in each layer. This could be done either in a deterministic way or stochastic way. We believe that training the network with such new methods of regularization has a potential to improve the performance after several epochs.

We also plan to apply deep architectures with particular use of optimized kernel machines to enable dealing with problems (and datasets) with limited data and prior knowledge of the relationships in

data. Enabling architectures with data-specific kernels or kernel compositions (like *multiple kernel learning*), we believe, would lead to new and powerful architectures and inference strategies.

Finally, as mentioned in section 4, the lack of very large amounts of training data encourages us to experiment with One-Shot approaches.

APPENDIX A: Verification

The following is a correspondence between the author and the editor of *The Journal of Experimental and Theoretical Artificial Intelligence* concerning the republishing into this thesis the original works found in [30]:

Dr. Dietrich,

My article titled ”A deep learning approach to diagnosing schizophrenic patients” is to be published in your journal (TETA-2018-0188.R2). I would like to use the same article for my master’s thesis.

Ideally, I’d like to do as little rewriting as possible, preferably no rewriting at all. What are the guidelines for this? Is there a standard procedure?

Respectfully,

Justin Barry

Yes, there is a standard procedure. In you MA thesis, you just say something like ”This thesis has previously been published in the Journal of Experimental and Theoretical AI ¡volume and issue number, page numbers, year¿.”

e

LIST OF REFERENCES

[1] Elena A Allen, Erik B Erhardt, Eswar Damaraju, William Gruner, Judith M Segall, Rogers F Silva, Martin Havlicek, Srinivas Rachakonda, Jill Fries, Ravi Kalyanam, et al. A baseline for the multivariate comparison of resting-state networks. *Frontiers in systems neuroscience*, 5:2, 2011.

[2] Javeria Amin, Muhammad Sharif, Mussarat Yasmin, and Steven Lawrence Fernandes. Big data analysis for brain tumor detection: Deep convolutional neural networks. *Future Generation Computer Systems*, 2018.

[3] Belhumeur. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on pattern analysis and machine intelligence*, 19(7):711–720, 1997.

[4] C Bois, HC Whalley, AM McIntosh, and SM Lawrie. Structural magnetic resonance imaging markers of susceptibility and transition to schizophrenia: A review of familial and clinical high risk population studies. *Journal of Psychopharmacology*, 29(2):144–154, 2015.

[5] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 144–152, New York, NY, USA, 1992. ACM.

[6] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.

[7] M. Cetin, F. Christensen, C. Abbott, J. Stephen, A. Mayer, J. Canive, J. Bustillo, G. Pearlson, , and V. D. Calhoun. Thalamus and posterior temporal lobe show greater inter-network connectivity at rest and across varying sensory loads in schizophrenia. Neuroimage, 2014.

[8] François Chollet et al. Keras. `https://keras.io`, 2015.

[9] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, Sep 1995.

[10] J.S. Cramer. The origins of logistic regression.

[11] Oguz Demirci and Vince D Calhoun. Functional magnetic resonance imaging–implications for detection of schizophrenia. *European neurological review*, 4(2):103, 2009.

[12] Erik Barry Erhardt, Srinivas Rachakonda, Edward J Bedrick, Elena A Allen, Tülay Adali, and Vince D Calhoun. Comparison of multi-subject ica methods for analysis of fmri data. *Human brain mapping*, 32(12):2075–2095, 2011.

[13] Steven Lawrence Fernandes and G Josemin Bala. Fusion of sparse representation and dictionary matching for identification of humans in uncontrolled environment. *Computers in biology and medicine*, 76:215–237, 2016.

[14] Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems? *Journal of Machine Learning Research*, 15:3133–3181, 2014.

[15] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *JMLR W&CP: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2010)*, volume 9, pages 249–256, May 2010.

[16] Shaoqiang Han, Wei Huang, Yan Zhang, Jingping Zhao, and Huafu Chen. Recognition of early-onset schizophrenia using deep-learning method. *Applied Informatics*, 4(1):16, Dec 2017.

[17] David W. Hosmer and Stanley Lemeshow. *Applied logistic regression (Wiley Series in probability and statistics)*. Wiley-Interscience Publication, 2 edition, 2000.

[18] Tsung-Hao Hsieh, Ming-Jian Sun, and Sheng-Fu Liang. Diagnosis of schizophrenia patients based on brain network complexity analysis of resting-state fmri. In *The 15th International Conference on Biomedical Engineering*, pages 203–206. Springer, 2014.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, pages 1097–1105, USA, 2012. Curran Associates Inc.

[20] Maarten PJ Kuenen, Massimo Mischi, and Hessel Wijkstra. Contrast-ultrasound diffusion imaging for localization of prostate cancer. *IEEE transactions on medical imaging*, 30(8):1493–1502, 2011.

[21] Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350:1332–1338, 12/11/2015 2015.

[22] Yann Lecun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 5 2015.

[23] Roshan J. Martis, Hong Lin, Varadraj P. Gurupur, and Steven L. Fernandes. Editorial: Frontiers in development of intelligent applications for medical imaging processing and computer vision. *Computers in Biology and Medicine*, 89:549 – 550, 2017.

[24] Cathrine Mihalopoulos, Meredith Harris, Lisa Henry, Susy Harrigan, and Patrick McGorry. Is early intervention in psychosis cost-effective over the long term? *Schizophrenia bulletin*, 35(5):909–918, 2009.

[25] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE transactions on pattern analysis and machine intelligence*, 27(10):1615–1630, 2005.

[26] Michael J. Owen, Akira Sawa, and Preben B. Mortensen. Schizophrenia. *The Lancet*, 388(10039):86–97, Jul 02 2016. Copyright - Copyright Elsevier Limited Jul 2, 2016; Last updated - 2017-11-23; CODEN - LANCAO.

[27] Pinkal Patel, Priya Aggarwal, and Anubha Gupta. Classification of schizophrenia versus normal subjects using deep learning. pages 1–6, 12 2016.

[28] Judith Segall, Elena Allen, Rex Jung, Erik Erhardt, Sunil Arja, Kent Kiehl, and Vince Calhoun. Correspondence between structure and function in the human brain at rest. *Frontiers in Neuroinformatics*, 6:10, 2012.

[29] Rogers Silva, Eduardo Castro, Cota Gupta, Mustafa Cetin, Mohammad Arbabshirani, Vamsi Potluru, Sergey Plis, and Vince Calhoun. The tenth annual mlsp competition: Schizophrenia classification challenge. 09 2014.

[30] Srivathsan Srinivasagopalan, Justin Barry, Varadraj Gurupur, and Sharma Thankachan. A deep learning approach for diagnosing schizophrenic patients. *Journal of Experimental & Theoretical Artificial Intelligence*, 0(0):1–14, 2019.

[31] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[32] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, Berlin, Heidelberg, 1995.

[33] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[34] Prashanthi Vemuri and Clifford R. Jack. Role of structural mri in alzheimer's disease. *Alzheimer's Research & Therapy*, 2(4):23, Aug 2010.

[35] Oriol Vinyals, Charles Blundell, Tim Lillicrap, koray kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3630–3638. Curran Associates, Inc., 2016.

[36] Huei-Fang Yang, Kevin Lin, and Chu-Song Chen. Supervised learning of semantics-preserving hash via deep convolutional neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 40(2):437–451, 2018.

[37] Ling-Li Zeng, Huaning Wang, Panpan Hu, Bo Yang, Weidan Pu, Hui Shen, Xingui Chen, Zhening Liu, Hong Yin, Qingrong Tan, et al. Multi-site diagnostic classification of schizophrenia using discriminant deep learning with functional connectivity mri. *EBioMedicine*, 30:74–85, 2018.